(2) 12

(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2004-29992
(P2004-29992A)
(43)公開日　平成16年1月29日(2004.1.29)

(51)Int.Cl.[7]	F I		テーマコード（参考）
G05B 23/02	G05B 23/02	302Y	5B042
G05B 19/048	G06F 1/04	302Z	5B045
G06F 1/04	G06F 9/46	330C	5B098
G06F 9/46	G06F 9/46	340B	5H220
G06F 11/30	G06F 11/30	320D	5H223

審査請求 未請求　請求項の数 14　OL　（全 12 頁）　最終頁に続く

(21)出願番号　特願2002-182424(P2002-182424)
(22)出願日　平成14年6月24日(2002.6.24)

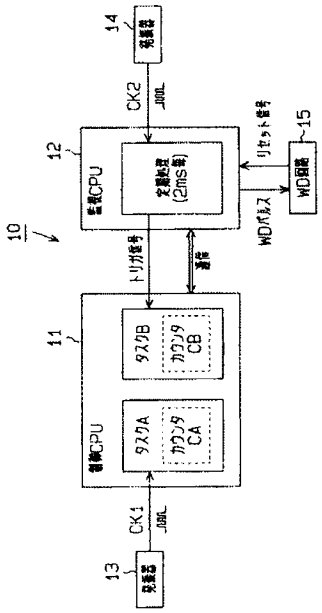
(71)出願人　000004260
株式会社デンソー
愛知県刈谷市昭和町１丁目１番地
(74)代理人　100068755
弁理士　恩田　博宣
(74)代理人　100105957
弁理士　恩田　誠
(72)発明者　宮地　和孝
愛知県刈谷市昭和町１丁目１番地　株式会社デンソー内
(72)発明者　小林　稔昌
愛知県刈谷市昭和町１丁目１番地　株式会社デンソー内

Fターム(参考)　5B042 JJ13 JJ15 JJ19 JJ21 JJ26 JJ29

最終頁に続く

(54)【発明の名称】電子制御装置

(57)【要約】
【課題】クロック異常や処理抜け異常を適正に検出し、ひいてはＣＰＵの信頼性の確保を図ること。
【解決手段】エンジンＥＣＵ１０は、制御ＣＰＵ１１と監視ＣＰＵ１２とを備える。監視ＣＰＵ１２は自身に入力されるクロックを基に一定周期のトリガ信号を生成し制御ＣＰＵ１１に送信する。また、制御ＣＰＵ１１は、自身に入力されるクロックを基に一定周期でタスクＡを起床し、カウンタＣＡをカウントすると共に、監視ＣＰＵ１２より入力されるトリガ信号を基にタスクＢを起床し、カウンタＣＢをカウントする。そして、これらカウンタＣＡ及びＣＢの差分を、前後する２つのタイミングで比較し、その比較結果からクロック異常を検出する。
【選択図】　図１



【特許請求の範囲】
【請求項1】
第１ＣＰＵと第２ＣＰＵとを備え、これら各ＣＰＵに個別のクロック信号が入力されると共に該クロック信号を基に各ＣＰＵで定期的な演算処理が実施され、前記第２ＣＰＵは自身に入力されるクロック信号を基に一定周期の時間情報を前記第１ＣＰＵに通知する電子制御装置であって、
前記第１ＣＰＵは、自身に入力されるクロック信号を基に一定周期で主カウンタをカウントすると共に、前記第２ＣＰＵより入力される一定周期の時間情報を基に副カウンタをカウントし、これら主副両カウンタの差分を、前後する２つのタイミングで比較し、その比較結果から前記クロック信号の異常を検出することを特徴とする電子制御装置。
【請求項2】
前記第１ＣＰＵは、一定周期でタスクＡを起床し、当該タスクＡにて主カウンタをカウントすると共に、前記第２ＣＰＵからの時間情報をトリガとしてタスクＢを起床し、当該タスクＢにて副カウンタをカウントする請求項１記載の電子制御装置。
【請求項3】
前記第２ＣＰＵは、クロック信号に基づく定期処理にて一定周期でトリガ信号を出力し、前記第１ＣＰＵは、該トリガ信号の入力の都度、副カウンタをカウントする請求項１又は２記載の電子制御装置。
【請求項4】
前記第２ＣＰＵは、クロック信号に基づく定期の通信処理にて一定周期でデータ送信を行い、前記第１ＣＰＵは、第２ＣＰＵからのデータ受信の都度、副カウンタをカウントする請求項１又は２記載の電子制御装置。
【請求項5】
前記第２ＣＰＵより所定周期で反転するウオッチドッグパルスを入力し、そのウオッチドッグパルスが所定時間以上反転しないと第２ＣＰＵに対してリセット信号を出力する監視回路を更に備え、前記第１ＣＰＵは、前記ウオッチドッグパルスを第２ＣＰＵより入力し、その入力の都度、副カウンタをカウントする請求項１又は２記載の電子制御装置。
【請求項6】
前記第１ＣＰＵは、前後２つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が毎回同一である場合に、クロック異常である旨判定する請求項１乃至５の何れかに記載の電子制御装置。
【請求項7】
前記第１ＣＰＵは、前後２つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が一方のカウンタのカウント分相当である場合に、何れかのカウンタが停止している旨判定する請求項１乃至６の何れかに記載の電子制御装置。
【請求項8】
主副両カウンタの周期が同一である請求項１乃至７の何れかに記載の電子制御装置。
【請求項9】
前記第１ＣＰＵは、前記第２ＣＰＵより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残し、次回のタスク起床時に前記履歴から処理抜け異常を検出する請求項１乃至８の何れかに記載の電子制御装置。
【請求項10】
請求項９記載の電子制御装置において、前記第１ＣＰＵは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残す電子制御装置。
【請求項11】
請求項９又は１０記載の電子制御装置において、前記第１ＣＰＵは、前記第２ＣＰＵからの時間情報にて起床される所定のタスクの最後に、前記副カウンタをカウントする電子制御装置。

【請求項12】
第１ＣＰＵと第２ＣＰＵとを備え、これら各ＣＰＵに個別のクロック信号が入力されると共に該クロック信号を基に各ＣＰＵで定期的な演算処理が実施され、前記第２ＣＰＵは自身に入力されるクロック信号を基に一定周期の時間情報を前記第１ＣＰＵに通知する電子制御装置であって、
前記第１ＣＰＵは、前記第２ＣＰＵより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残し、次回のタスク起床時に前記履歴から処理抜け異常を検出することを特徴とする電子制御装置。
【請求項13】
請求項１２記載の電子制御装置において、前記第１ＣＰＵは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残す電子制御装置。
【請求項14】
前記第１ＣＰＵは電子スロットル制御を実施し、前記第２ＣＰＵは電子スロットル制御に関するフェイルセーフ処理を実施する車両用電子制御装置として適用され、前記第１ＣＰＵは、クロック異常又は処理抜け異常である旨を検出した時にそれを第２ＣＰＵに通知し、第２ＣＰＵは、第１ＣＰＵからの異常情報に基づきフェイルセーフ処理を実施する請求項１乃至１３の何れかに記載の電子制御装置。

【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、２つ以上のＣＰＵを有する電子制御装置にかかり、特に各ＣＰＵに供給されるクロック信号の異常や処理抜け異常を好適に検出するための電子制御装置に関するものである。
【０００２】
【従来の技術】
車両制御等を実施する電子制御装置として、複数のＣＰＵを備え、各ＣＰＵ間で相互に異常を監視する機能を持つものがある。その一つとして、所定周期で反転するウオッチドッグパルス（ＷＤパルス）をモニタしてＣＰＵの動作状態を監視する方法がある。これは、一方のＣＰＵが、他方のＣＰＵよりＷＤパルスを入力し、該ＷＤパルスが停止すると当該他方のＣＰＵが異常である（暴走している）と判定するものである。
【０００３】
【発明が解決しようとする課題】
この種の電子制御装置に関わる異常として、発振器等から各ＣＰＵに供給されるクロック信号の異常がある。この場合、発振器の異常等によりクロック信号の周波数が大小変化すると、そのクロック異常の状態によってはその異常が検出できないことが考えられる。
【０００４】
例えば、クロック信号の周波数が正常時よりも大きくなる場合、それに起因してＣＰＵ内での時間演算の精度が低下する。そのため、定期的に実施されるべき各種の演算処理が所定の周期で実施されなくなるという不都合が生じる。かかる場合、上記の如くＷＤパルスによりＣＰＵの異常検出を実施しても、ＷＤパルスは定期的に発行されているため、クロック異常を判定することができない。
【０００５】
また、優先度の異なる複数のタスクを起床するＣＰＵでは、ＣＰＵの処理負荷が高まると低優先度のタスクで処理抜けが発生する場合が考えられる。この場合、通常ＷＤパルスの発行は高優先度の処理であるため、低優先度処理が抜けたときでも正常にＷＤパルスが発行される。そのため、処理抜けが発生してもＷＤパルスには影響が出ず、処理抜けを検出することができない。最悪の場合、そのまま処理を継続する可能性が考えられる。
【０００６】
本発明は上記問題に着目してなされたものであって、その目的とするところは、クロック

異常や処理抜け異常を適正に検出し、ひいてはＣＰＵの信頼性の確保を図ることができる電子制御装置を提供することである。

【０００７】

【課題を解決するための手段】

請求項１に記載の発明では、第１ＣＰＵと第２ＣＰＵとを備え、これら各ＣＰＵに個別のクロック信号が入力されると共に該クロック信号を基に各ＣＰＵで定期的な演算処理が実施される。前記第２ＣＰＵは自身に入力されるクロック信号を基に一定周期の時間情報を前記第１ＣＰＵに通知する。また特に、前記第１ＣＰＵは、自身に入力されるクロック信号を基に一定周期で主カウンタをカウントすると共に、前記第２ＣＰＵより入力される一定周期の時間情報を基に副カウンタをカウントする。そして、これら主副両カウンタの差分を、前後する２つのタイミングで比較し、その比較結果から前記クロック信号の異常を検出する。

【０００８】

仮に第２ＣＰＵ側のクロック信号が異常となり、その周波数が正常時よりも小さくなる場合、第２ＣＰＵから第１ＣＰＵへ通知される時間情報の時間間隔が広がり、結果として副カウンタのカウント動作が遅くなる。この場合、主副両カウンタの差分が正常時とは相違し、更に前後する差分を比較することで、クロック信号が異常である旨判定できる。つまり、クロック信号の周期が変化するような異常を検出することができる。その結果、クロック異常を適正に検出し、ひいてはＣＰＵの信頼性の確保を図ることができるようになる。

【０００９】

請求項２に記載の発明では、前記第１ＣＰＵは、一定周期でタスクＡを起床し、当該タスクＡにて主カウンタをカウントすると共に、前記第２ＣＰＵからの時間情報をトリガとしてタスクＢを起床し、当該タスクＢにて副カウンタをカウントする。この場合、タスクＡ及びタスクＢが適宜起床されることで主副両カウンタのカウント動作を行わせることができる。

【００１０】

上記の如く第１ＣＰＵは、第２ＣＰＵから時間情報の通知を受けて副カウンタを動作させるが、かかる場合において、時間情報の通知を以下の形態で実現すると良い。

・請求項３に記載の発明では、前記第２ＣＰＵは、クロック信号に基づく定期処理にて一定周期でトリガ信号を出力し、前記第１ＣＰＵは、該トリガ信号の入力の都度、副カウンタをカウントする。

・請求項４に記載の発明では、前記第２ＣＰＵは、クロック信号に基づく定期の通信処理にて一定周期でデータ送信を行い、前記第１ＣＰＵは、第２ＣＰＵからのデータ受信の都度、副カウンタをカウントする。

・請求項５に記載の発明では、前記第２ＣＰＵより所定周期で反転するウオッチドッグパルスを入力し、そのウオッチドッグパルスが所定時間以上反転しないと第２ＣＰＵに対してリセット信号を出力する監視回路を更に備え、前記第１ＣＰＵは、前記ウオッチドッグパルスを第２ＣＰＵより入力し、その入力の都度、副カウンタをカウントする。

【００１１】

上記何れの場合にも、第１ＣＰＵにおいて一定周期で副カウンタがカウントされ、その副カウンタと主カウンタとの差分によりクロック異常が適正に検出できる。

【００１２】

請求項６に記載の発明では、前記第１ＣＰＵは、前後２つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が毎回同一である場合に、クロック異常である旨判定する。これにより、クロック異常が精度良く検出できる。

【００１３】

請求項７に記載の発明では、前記第１ＣＰＵは、前後２つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が一方のカウンタのカウント分相当である場合に、何れかのカウンタが停止している旨判定する。これにより、カウンタ停止

の異常が精度良く検出できる。請求項２のように、タスクＡで主カウンタがカウントされ、タスクＢで副カウンタがカウントされる場合、何れかのタスクが停止したことが精度良く検出できる。

【００１４】

請求項８に記載したように、主副両カウンタの周期が同一である場合、それら両カウンタが共に正常であれば、差分が常に一致する。それ故、主副両カウンタの差分の比較が容易に実施できる。

【００１５】

請求項９に記載の発明では、前記第１ＣＰＵは、前記第２ＣＰＵより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残す。そして、次回のタスク起床時に前記履歴から処理抜け異常を検出する。この場合、前記タスクが最後まで実施されれば、全処理について処理終了の履歴が残るが、途中までしか実施されなければ未実施分については処理終了の履歴が残らない。従って、当該タスクについて処理抜けを検出することができる。また、履歴を確認することにより、処理抜けが発生した位置を特定することができる。

【００１６】

上記請求項９の発明では請求項１０に記載したように、前記第１ＣＰＵは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残すと良い。優先度の異なる複数のタスクが存在する場合、優先度の低いタスクは、その実施途中で高優先度のタスクが割り込むことで処理抜け（処理の中断）が生じるおそれがあるが、かかる場合に処理抜けの発生やその発生場所の特定が可能となる。

【００１７】

また、請求項１１に記載したように、前記第１ＣＰＵは、前記第２ＣＰＵからの時間情報にて起床される所定のタスクの最後に、前記副カウンタをカウントすると良い。これにより、処理抜けが発生した場合に副カウンタがカウントされなくなり、主副両カウンタの差分から異常発生の判定が可能となる。

【００１８】

一方、請求項１２に記載の発明でも、上記請求項９と同様に、前記タスクが最後まで実施されれば、全処理について処理終了の履歴が残るが、途中までしか実施されなければ未実施分については処理終了の履歴が残らない。従って、当該タスクについて処理抜けを検出することができる。また、履歴を確認することにより、処理抜けが発生した位置を特定することができる。

【００１９】

更に、請求項１３に記載したように、前記第１ＣＰＵは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残すことで、高優先度のタスクが割り込んで処理抜け（処理の中断）が発生しても、当該処理抜けの発生やその発生場所の特定が可能となる。

【００２０】

また、請求項１４に記載の発明は、前記第１ＣＰＵは電子スロットル制御を実施し、前記第２ＣＰＵは電子スロットル制御に関するフェイルセーフ処理を実施する車両用電子制御装置として適用される。そして、前記第１ＣＰＵは、クロック異常又は処理抜け異常である旨を検出した時にそれを第２ＣＰＵに通知し、第２ＣＰＵは、第１ＣＰＵからの異常情報に基づきフェイルセーフ処理を実施する。この場合、クロック異常や処理抜け異常が発生しても、その適正なフェイルセーフ処理が実施できる。

【００２１】

【発明の実施の形態】

以下、本発明を具体化した一実施の形態を図面に従って説明する。本実施の形態では、車両に搭載されるエンジンＥＣＵとして本発明の電子制御装置を具体化している。図１は、本実施の形態におけるエンジンＥＣＵの構成を示すブロック図である。

【００２２】

図１において、エンジンＥＣＵ１０は、エンジンの噴射制御、点火制御及び電子スロットル制御を実施するための制御ＣＰＵ１１と、電子スロットル制御に関するフェイルセーフ処理や制御ＣＰＵ１１の監視を実施するための監視ＣＰＵ１２とを備える。これらの各ＣＰＵ１１，１２は相互に通信可能に接続されている。制御ＣＰＵ１１には発振器１３よりクロックＣＫ１（クロック信号）が入力され、監視ＣＰＵ１２には発振器１４よりクロックＣＫ２（クロック信号）が入力される。なお本実施の形態では、制御ＣＰＵ１１が「第１ＣＰＵ」に相当し、監視ＣＰＵ１２が「第２ＣＰＵ」に相当する。

【００２３】

制御ＣＰＵ１１は、発振器１３からのクロックＣＫ１に基づき一定周期でタスクＡを起床し、そのタスクＡ内にて毎回カウンタＣＡをカウントアップする。タスクＡは高優先度のタスクであり、本実施の形態では２ｍｓ毎に起床されるようになっている。また、制御ＣＰＵ１１には監視ＣＰＵ１２からトリガ信号が取り込まれる。制御ＣＰＵ１１はこのトリガ信号に基づきタスクＢを起床し、そのタスクＢ内にて毎回カウンタＣＢをカウントアップする。

【００２４】

一方、監視ＣＰＵ１２は、発振器１４からのクロックＣＫ２に基づき２ｍｓ毎に定期処理を起床し、その定期処理にて一定周期のトリガ信号を生成し出力する。このトリガ信号が制御ＣＰＵ１１に取り込まれる。本実施の形態では、カウンタＣＡが「主カウンタ」に、カウンタＣＢが「副カウンタ」に相当し、トリガ信号が「一定周期の時間情報」に相当する。

【００２５】

また、監視ＣＰＵ１２は「監視回路」としてのＷＤ回路１５に対してＷＤパルスを出力し、ＷＤ回路１５は監視ＣＰＵ１２からのＷＤパルスが所定時間以上反転しなかった場合に監視ＣＰＵ１２に対してリセット信号を出力する。

【００２６】

ここで、カウンタＣＡ及びＣＢの動作を図２にて説明する。カウンタＣＡ及びＣＢが共に正常動作する場合、図２の（ａ）に示すように、各カウンタが何れも２ｍｓ毎にカウントアップされる。この場合、カウンタＣＡ及びＣＢの差分をΔＣｎ　（＝ＣＡ－ＣＢ）とすると、タイミングｔ１ではその差分がΔＣ１　、タイミングｔ２ではその差分がΔＣ２　となる。図２の（ａ）は正常動作を表すためΔＣ１　＝ΔＣ２　となる。

【００２７】

これに対し、図２の（ｂ）～（ｄ）はカウンタＣＢが異常動作となる場合を示す。（ｂ）～（ｄ）を詳しく説明する。（ｂ）は、カウンタＣＢの周期が正常時よりも大きくなっている。これは監視ＣＰＵ１２側のクロックＣＫ２が異常（クロック異常）となり、それが原因で定期処理の起床周期（トリガ信号の周期）が変わり、更にはカウンタＣＢの周期が変化したと考えられる。この場合、タイミングｔ１，ｔ２での差分ΔＣ１　，ΔＣ２　はΔＣ１　＜ΔＣ２　となる。

【００２８】

また、（ｃ）は、カウンタＣＢが停止する異常を示し、（ｄ）は、監視ＣＰＵ１２での定期処理が抜けてしまう異常（処理抜け異常）を示す。なお、処理抜けは不定期に発生し、この処理抜けによりカウンタＣＢが一定周期でカウントアップされない事態が生じる。これら（ｃ），（ｄ）の場合にもやはり、タイミングｔ１，ｔ２での差分ΔＣ１　，ΔＣ２　がΔＣ１　＜ΔＣ２　となる。

【００２９】

本実施の形態では、上記図２の（ｂ）～（ｄ）の各異常について何れも検出可能であって、更に各異常の形態を特定できる異常検出手法を提案する。上記図２の（ｂ）の場合、カウンタＣＡ及びＣＢのカウントアップの周期が相違するため各カウンタ値の差分ΔＣｎ　は毎回相違するが、同ΔＣｎ　の変化は一律であり、一定の時間間隔で前後する２つのΔＣｎ　を比較すると、その変化量は一定である。つまり、前後する差分の変化量をＤＣｎ　（以下、差分変化量ＤＣｎ　）とすると、その差分変化量ＤＣｎ　は毎回同じ値となる

（ＤＣｎ　＝ＤＣｎ－１　となる）。それ故、その差分変化量ＤＣｎ　によりクロック異常が特定できる。勿論、カウンタＣＢの周期が小さくなる場合や、カウンタＣＡの周期が大きくなる（又は小さくなる）場合にも同様にクロック異常が特定できる。

【００３０】

また、上記図２の（ｃ）の場合、一方のカウンタＣＡは正常にカウントアップし、他方のカウンタＣＢは停止しているため、各カウンタＣＡ及びＣＢの差分変化量ＤＣｎ　はカウンタＣＡのカウントアップ分に相当する。この場合、差分変化量ＤＣｎ　は、ｔ１－ｔ２間の時間間隔ＤＴｎ　をカウンタ１周期分の時間Ｔ（２ｍｓ）で除算した値（ＤＴｎ　／Ｔ）に一致する（ＤＣｎ　＝ＤＴｎ　／Ｔとなる）。これにより、タスクＢ停止の異常が特定できる。

【００３１】

また、上記図２の（ｄ）の場合、上記（ｂ），（ｃ）と同様にΔＣ１　≠ΔＣ２　（すなわち、ＤＣｎ　≠０）であるが、差分ΔＣｎ　の変化は不定期なものとなる。よって、ΔＣ１　≠ΔＣ２　であり、且つ上記図２の（ｂ），（ｃ）に該当しなければ、処理抜け異常であると特定できる。

【００３２】

次に、各ＣＰＵ１１，１２で起床される演算処理について図３～図６のフローチャートを参照しながら詳しく説明する。先ずはじめに、図３はタスクＡの手順を示すフローチャートであり、このタスクＡは制御ＣＰＵ１１により２ｍｓ毎に起床される。

【００３３】

図３において、先ずステップ１０１では、カウンタＣＡ及びＣＢに基づいて異常検出処理を実施する。但しその詳細（図４の処理）は後述する。また、ステップ１０２では、タスクＡに関して通常処理を実施し、続くステップ１０３では、カウンタＣＡを１インクリメントする。このタスクＡにより、２ｍｓ毎にカウンタＣＡがカウントアップされるようになる。

【００３４】

図４の異常検出処理では、８ｍｓ毎に一連の異常検出を実施することとしており、ステップ２０１がＹＥＳであることを条件に後続のステップ２０２に進む（つまり、タスクＡ起床の４回に１回の割合で異常検出が実施される）。

【００３５】

ステップ２０２に進むと、カウンタＣＡ及びＣＢの差分ΔＣｎ　がどれだけ変化したかを示す差分変化量ＤＣｎ　を算出する。具体的には、カウンタＣＡ及びＣＢの差分の今回値ΔＣｎ　と前回値ΔＣｎ－１　との差を取り、差分変化量ＤＣｎ　を算出する（ＤＣｎ　＝ΔＣｎ　－ΔＣｎ－１　）。

【００３６】

その後、ステップ２０３では、差分変化量ＤＣｎ　が０であるか否かを判別する。前記図２の（ａ）で説明した通りカウンタＣＡ及びＣＢが何れも正常の場合、ＤＣｎ　＝０（ΔＣｎ　＝ΔＣｎ－１　）となる。ＤＣｎ　＝０の場合、ステップ２０４に進み、各種の異常カウンタＥｒｒＣｋ，ＥｒｒＦ，ＥｒｒＳｔを何れも０にクリアする。そしてその後、本処理を一旦終了する。

【００３７】

また、ＤＣｎ≠０の場合、ステップ２０５に進み、差分変化量の今回値ＤＣｎ　と前回値ＤＣｎ－１　とを比較する。そして、ＤＣｎ　＝ＤＣｎ－１　であれば、クロック異常であると推定し、ステップ２０６でクロック異常カウンタＥｒｒＣｋを１インクリメントする。これは、前記図２の（ｂ）の場合に該当する。

【００３８】

また、ＤＣｎ　≠ＤＣｎ－１　の場合、ステップ２０７で時間間隔ＤＴｎ　（前回の異常検出から今回の異常検出までの時間幅）を算出し、続くステップ２０８でＤＣｎ　＝ＤＴｎ　／Ｔであるか否かを判別する。そして、ＹＥＳの場合、タスクＢの停止異常であると推定し、ステップ２０９でタスクＢ停止カウンタＥｒｒＳｔを１インクリメントする。ま

た、ＮＯの場合、処理抜け異常であると推定し、ステップ２１０で処理抜け異常カウンタＥｒｒＦを１インクリメントする。ステップ２０８がＹＥＳであることは前記図２の（ｃ）の場合に該当し、同ステップ２０８がＮＯであることは前記図２の（ｄ）の場合に該当する。

【００３９】

その後、ステップ２１１～２１３では、各異常カウンタＥｒｒＣｋ，ＥｒｒＳｔ，ＥｒｒＦが所定の判定値Ｌｍｔよりも小さいか否かを判別する。そして、何れも判定値Ｌｍｔ未満であれば、そのまま本処理を終了する。また、判定値Ｌｍｔ以上となる異常については、それに対応する異常フラグ（ＸＥｒｒＣｋ，ＸＥｒｒＳｔ，ＸＥｒｒＦの何れか）をＯＮする（ステップ２１４～２１６）。なお、判定値Ｌｍｔは、何れも同一にする必要はなく、各異常カウンタＥｒｒＣｋ，ＥｒｒＳｔ，ＥｒｒＦ毎に個別に設定することも可能である。また、同一の異常が所定回連続して検出された場合にのみ、異常フラグをＯＮする構成としても良い。

【００４０】

また、図５はタスクＢの手順を示すフローチャートであり、このタスクＢは、監視ＣＰＵ１２からトリガ信号を入力する都度制御ＣＰＵ１１により起床される。実際には、トリガ信号は２ｍｓ周期のパルス信号であり、タスクＡと同じ周期でこのタスクＢが実施されるようになっている。因みに、トリガ信号は、図６に示す定期処理にて生成される。つまり、監視ＣＰＵ１２は、定期処理内の通常処理（ステップ４０１，４０３）の途中でトリガ信号を生成し出力する（ステップ４０２）。

【００４１】

タスクＢでは、ｎ個の処理（処理１～処理ｎ）が順次実施され、各処理が実施される都度、処理完了を表す処理完了フラグＸｆｉｎがＯＮされる。つまりこのとき、処理１～処理ｎが規定の処理単位であり、その処理終了の履歴が処理完了フラグＸｆｉｎとして残される。そして、次回のタスクＢ起床時においてこの処理完了フラグＸｆｉｎに基づき、処理抜け場所が特定されるようになっている。

【００４２】

図５において、先ずステップ３０１では、処理番号を示すｉを０にクリアし、続くステップ３０２では、処理ｉの処理完了フラグｘｆｉｎ（ｉ）がＯＮしているか否かを判別する。フラグＯＮであれば、前回のタスクＢで当該処理ｉが処理完了したと判断される。この場合、ステップ３０３でｉ＝ｎであるか否かを判別する。そして、ｉ＝ｎが成立するまで、ステップ３０４でｉを１ずつ加算しつつ、フラグｘｆｉｎ（ｉ）がＯＮしているか否かを繰り返し判別する。

【００４３】

フラグｘｆｉｎ（ｉ）＝ＯＦＦの場合には、その該当する処理ｉで処理抜けが発生したと判断できる。そのため、ステップ３０５に進み、その時の処理ｉを処理抜け発生位置としてメモリに記憶する。

【００４４】

その後、ステップ３０６では、処理完了フラグｘｆｉｎを全てクリアする。ステップ３０７～３１２では、処理１～処理ｎを順次実施すると共に、各処理の実施直後に、各々に対応する処理完了フラグＸｆｉｎをＯＮしていく。最後に、ステップ３１３では、カウンタＣＢを１インクリメントする。

【００４５】

上記図４の異常検出処理にて異常である旨検出されると（前記異常フラグがＯＮされると）、それが制御ＣＰＵ１１から監視ＣＰＵ１２に通知され、監視ＣＰＵ１２では、電子スロットル制御の通電カットなど、所定のフェイルセーフ処理が実施される。この場合、クロック異常や処理抜け異常が発生しても、その適正なフェイルセーフ処理が実施でき、エンジンＥＣＵ１０としての信頼性が確保される。

【００４６】

以上詳述した本実施の形態によれば、以下に示す効果が得られる。

クロック異常が発生すると、カウンタＣＡ及びＣＢの差分ΔＣｎ　が正常時とは相違するため、前後する２つの差分ΔＣｎ　を比較することでクロック異常である旨判定できる。つまり、クロックの周期が変化するような異常を検出することができる。その結果、クロック異常を適正に検出し、ひいてはＣＰＵの信頼性の確保を図ることができるようになる。

【００４７】

カウンタＣＡ及びＣＢの差分変化量ＤＣｎ　が毎回同一である場合にクロック異常である旨判定するため、クロック異常が精度良く検出できる。

また、カウンタＣＡ及びＣＢの差分変化量ＤＣｎ　が一方のカウンタのカウント分相当である場合に何れかのカウンタが停止している旨判定するため、カウンタ停止の異常が精度良く検出できる。またこれは、何れかのタスクが停止したことが精度良く検出できることと同意である。

【００４８】

タスクＢ内において規定の処理単位で処理終了の履歴（処理完了フラグｘｆｉｎ）を残し、次回のタスク起床時に前記履歴から処理抜け異常を検出するため、処理抜けが発生した位置を確実に特定することができる。

【００４９】

なお本発明は、上記以外に次の形態にて具体化できる。

上記実施の形態では、制御ＣＰＵ１１は、監視ＣＰＵ１２からトリガ信号を入力し、そのトリガ信号にてタスクＢを起床したが、この構成を変更しても良い。例えば、

（１）制御ＣＰＵ１１は、監視ＣＰＵ１２から定期的にデータを受信する都度、タスクＢを起床する。但しこの場合、上記２つのＣＰＵ間で定期的な通信が行われることが前提条件となる。

（２）制御ＣＰＵ１１は、監視ＣＰＵ１２からＷＤパルスを入力し、その入力の都度、タスクＢを起床する。この場合、前記図１の構成において、監視ＣＰＵ１２からＷＤ回路１５へのＷＤパルスを分岐させ、当該ＷＤパルスを制御ＣＰＵ１１にも入力する構成とすれば良い。

【００５０】

これら何れの場合にも、制御ＣＰＵ１１において一定周期でカウンタＣＢがカウントされ、そのカウンタＣＡ及びＣＢの差分によりクロック異常が適正に検出できる。

【００５１】

上記実施の形態では、カウンタＣＡ及びＣＢの周期（すなわち、タスクＡ及びタスクＢの起床周期）が同一である場合を例示したが、その周期が相違するものであっても良い。カウンタＣＡ及びＣＢの周期が相違しても、各カウンタが何れも正常であればその差分ΔＣｎ　はある規則性を保つ。そのため、やはり前後する２つのタイミングで差分ΔＣｎ　を比較し、その比較結果からクロック異常が検出できる。

【００５２】

本発明は、エンジンＥＣＵ以外にも他のＥＣＵに適用でき、更に車両用電子制御装置以外の用途に具体化することも可能である。要は、２つ以上のＣＰＵを備え、各ＣＰＵに個別のクロック信号が入力されると共に該クロック信号を基に各ＣＰＵで定期的な演算処理が実施される構成の電子制御装置であれば本発明が適用できる。

【図面の簡単な説明】

【図１】発明の実施の形態におけるエンジンＥＣＵの構成を示すブロック図。

【図２】（ａ）～（ｄ）はカウンタ動作を示すタイムチャート。

【図３】制御ＣＰＵによるタスクＡの手順を示すフローチャート。

【図４】制御ＣＰＵによる異常検出処理を示すフローチャート。

【図５】制御ＣＰＵによるタスクＢの手順を示すフローチャート。

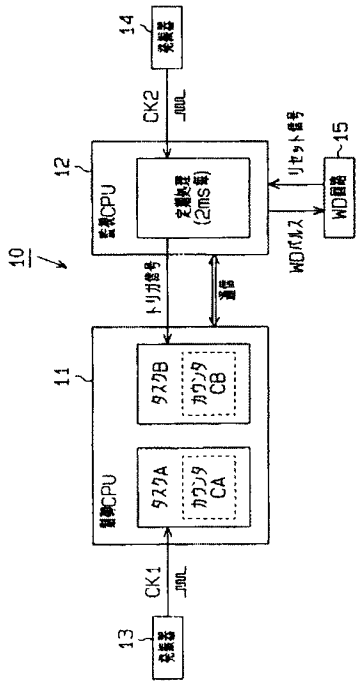
【図６】監視ＣＰＵによる定期処理を示すフローチャート。

【符号の説明】

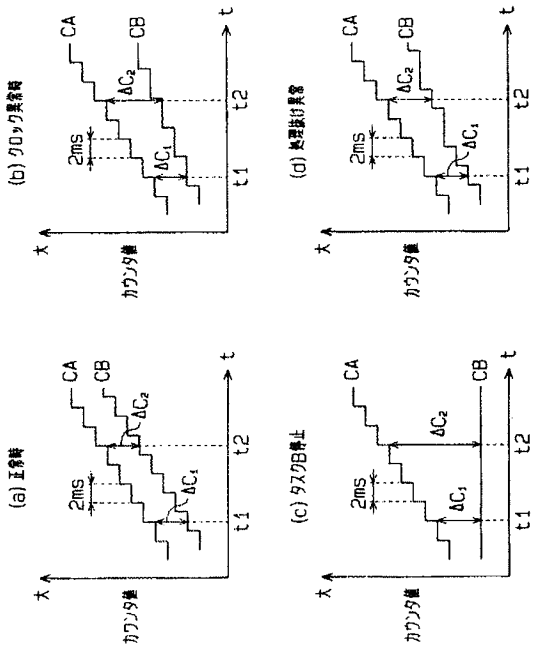
１０…ＥＣＵ、１１…制御ＣＰＵ、１２…監視ＣＰＵ、１３，１４…発振器、１５…監視

回路。

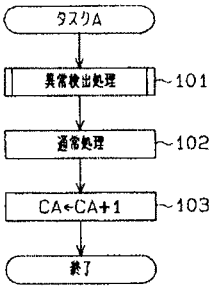
【図1】


10
11
12
13
14
15
CK1
CK2
タスクA
カウンタ CA
タスクB
カウンタ CB
トリガ信号
WDパルス
リセット信号
WD回路

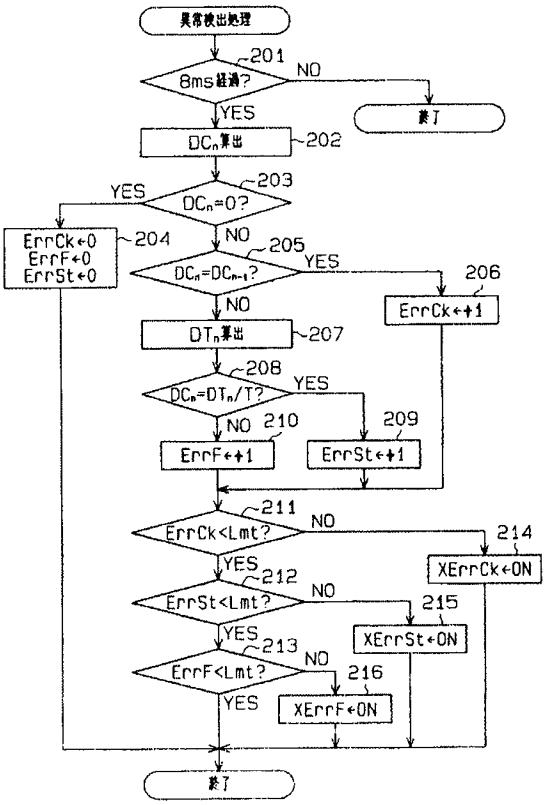
【図2】


(a) 正常時
(b) クロック異常時
(c) タスクB停止
(d) 処理遅れ異常
CA
CB
2ms
ΔC1
ΔC2
t1
t2
t
大
カウンタ値

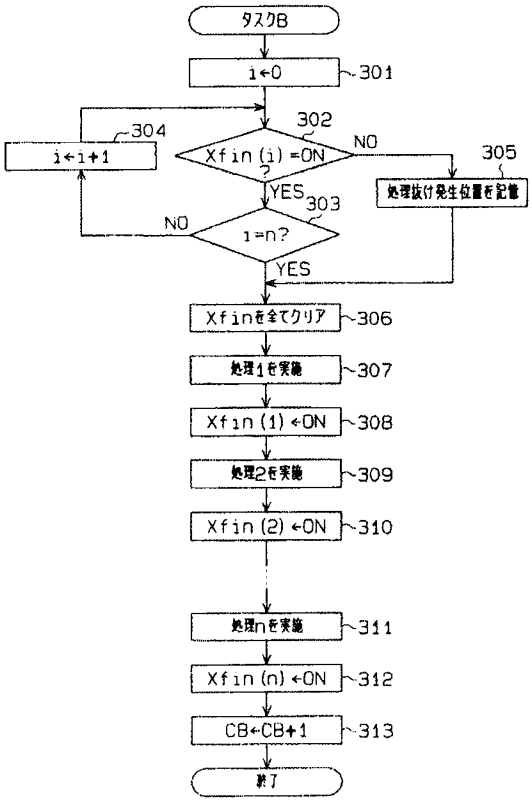
【図3】


タスクA
異常検出処理
101
通常処理
102
CA←CA+1
103
終了

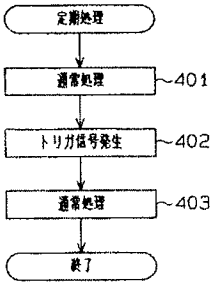
【図4】


異常検出処理
201
8ms経過?
NO
YES
終了
DCn算出
202
203
DCn=0?
YES
NO
ErrCk←0
ErrF←0
ErrSt←0
204
205
DCn=DCn-1?
YES
206
ErrCk←+1
NO
DTn算出
207
208
DCn=DTn/T?
YES
NO
209
210
ErrF←+1
ErrSt←+1
211
ErrCk<Lmt?
NO
214
XErrCk←ON
YES
212
ErrSt<Lmt?
NO
215
XErrSt←ON
YES
213
ErrF<Lmt?
NO
216
XErrF←ON
YES
終了

【図5】


タスクB
i←0
301
302
304
i←i+1
Xfin(i)=ON
?
NO
305
処理抜け発生位置を記憶
YES
303
NO
i=n?
YES
Xfinを全てクリア
306
処理1を実施
307
Xfin(1)←ON
308
処理2を実施
309
Xfin(2)←ON
310
処理nを実施
311
Xfin(n)←ON
312
CB←CB+1
313
終了

【図6】


定期処理
通常処理
401
トリガ信号発生
402
通常処理
403
終了

(51)Int.Cl.[7]	F I	テーマコード（参考）
Ｇ０６Ｆ　15/177	Ｇ０６Ｆ　15/177　６７８Ａ	
	Ｇ０５Ｂ　19/05　Ｄ	

Ｆターム(参考)　5B045 JJ02 JJ13
5B098 AA10 BA04 GA02 GA04 GC06
5H220 BB09 CC07 CC09 CX01 EE10 HH03 JJ12 JJ26 KK01 LL01
5H223 AA10 CC03 DD03 EE04